EE/CprE/SE 492 WEEKLY REPORT 4

Oct 4th - Oct 31st

Group number: **20**

Project title: **GridGPT**

Client &/Advisor: **Gelli Ravikumar**

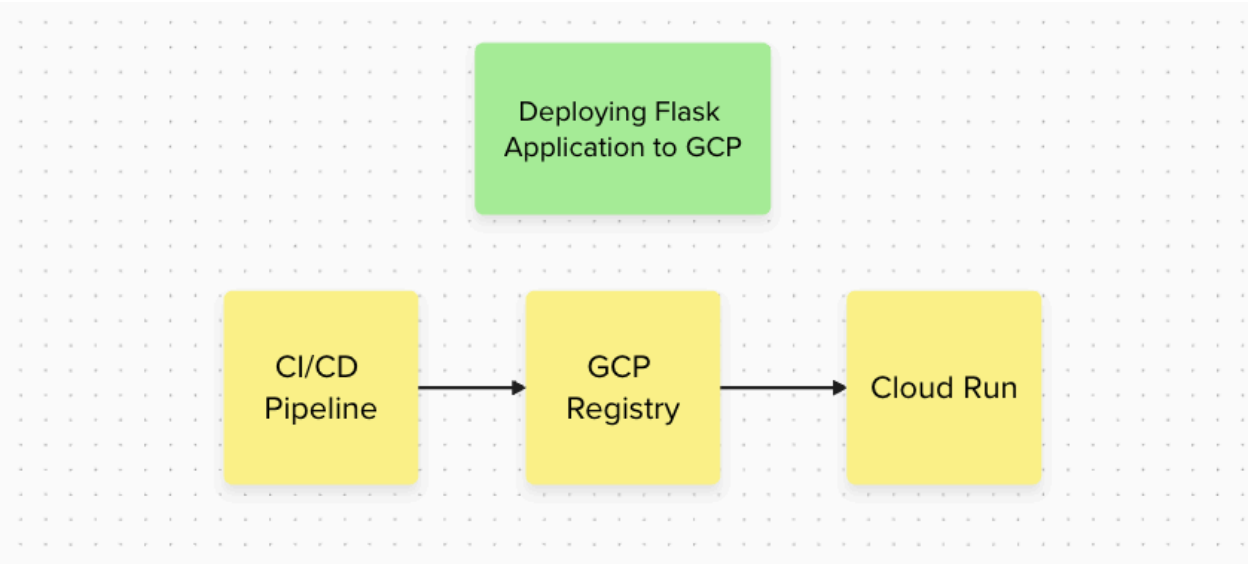
Team Members/Role:

- **Tin Ngo -> AI Integration Specialist/Backend Lead**
- **Jackson Phillips -> AI Integration Specialist**
- **Emma Heithoff -> Power Systems Specialist**
- **Eddy Andrade -> Frontend Lead**
- **Nick Doty -> Power Systems Specialist**

- **Weekly Summary**

- **Past week accomplishments**

- Tin Ngo: I created an outline of setting up the pipeline to deploy to GCP and get this to be a working API. In order to do so, I need to create a CI/CD pipeline that containerizes the application that I had already created and uploads it to the GCP Google Cloud Container Registry. I will use Cloud Build which is Google's CI/CD service that will make deploying this container easy. After the container is uploaded, I can deploy the container to Cloud Run. From there, we can call Cloud run via API endpoint and successfully use HuggingFace. In order to complete all this, I needed to read up a lot on GCP. I have not looked into the cost of it all but I know that GCP Cloud Run is serverless, so we will only pay for what we use which will be very small. We also pay for the space we take up in the GCP registry.

- 

- ai > huggingFace > Dockerfile > ...

```dockerfile
# Use an official Python runtime as a parent image
FROM python:3.11-slim

# Set the working directory to /app
WORKDIR /app

# Copy the requirements file
COPY src/requirements.txt .

# Install the dependencies
RUN pip install --no-cache-dir -r requirements.txt

# Copy the application code
COPY src/ .

COPY helper_script.sh .
RUN chmod +x helper_script.sh

# Run the command to start the application when the container launches
CMD ["./helper_script.sh"]
```

- ai > huggingFace > $ helper_script.sh

```
#!/usr/bin/env bash

# for debugging
# catch errors early and pick up the last error that occurred before exiting
set -eo pipefail

# gunicorn is the service that deploys the Flask app
# leave $PORT as an environment variable, never hardcode it, and GCP will pick up the port name
# main:app main is the name of the Python file that contains Flask app (main.py)
# app is the Flask app name I specified in main.py
exec gunicorn --bind :$PORT --workers 1 --threads 8 --timeout 0 main:app

# Exit immediately when one of the background processes terminate.
wait -n
```

```yaml
# cloudbuild.yaml
steps:
  # Build the container image
  - name: 'gcr.io/cloud-builders/docker'
    args: [
      'build',
      '-t', 'gcr.io/$PROJECT_ID/$REPO_NAME:$COMMIT_SHA',
      '-f', 'ai/huggingFace/Dockerfile',
      'ai/huggingFace'
    ]

  # Push the container image to Container Registry
  - name: 'gcr.io/cloud-builders/docker'
    args: ['push', 'gcr.io/$PROJECT_ID/$REPO_NAME:$COMMIT_SHA']

  # Deploy container image to Cloud Run
  - name: 'gcr.io/google.com/cloudsdktool/cloud-sdk'
    entrypoint: gcloud
    args:
    - 'run'
    - 'deploy'
    - '$REPO_NAME'
    - '--image'
    - 'gcr.io/$PROJECT_ID/$REPO_NAME:$COMMIT_SHA'
    - '--region'
    - 'us-central1'
    - '--platform'
    - 'managed'
    - '--allow-unauthenticated'

images:
  - 'gcr.io/$PROJECT_ID/$REPO_NAME:$COMMIT_SHA'
```

Week 2

After meeting with Professor Gelli, I took his feedback and decided to shift my focus to more functionality, instead of getting ready for deployment. I worked closely with Emma Heithoff to work towards generating questions and answers we should expect a grid operator to ask an LLM. I made a lot of code changes in preparation to make a proof of concept. This POC should demonstrate the ability to give valuable feedback to the operator based on the data in the DSS files. I worked on prompt engineering and feeding the openAI with background information with the data in the dss files. So far we have seen success in our answers.

So far we have these questions and expected answers:

1. Question: What is the voltage rating of Load S1A?

   A: 2.4 kV

```
(venv) tinngo@Tins-MacBook-Pro sddec24-20 % curl -X POST http://127.0.0.1:5000/generate_text \
-H "Content-Type: application/json" \
-d '{
  "prompt": "What is the voltage rating of Load S1A?",
  "model": "gpt-3.5-turbo-0125",
  "context_directory": "grid_applications/distribution_grid/grid_dss/src/systems/123Bus"
}'
data:

Final Summary:


data: The voltage rating of Load S1A is 2.4 kV for Wye connection.

The voltage rating of Load S1A is 2.4 kV.

The information about the voltage rating of Load S1A is not provided in the given context.

data: [DONE]

(venv) tinngo@Tins-MacBook-Pro sddec24-20 %
```

2. Q: What bus is Load S1A connected to?

   A: Bus 1

```
(venv) tinngo@Tins-MacBook-Pro sddec24-20 % curl -X POST http://127.0.0.1:5000/generate_text \
-H "Content-Type: application/json" \
-d '{
  "prompt": "What bus is Load S1A connected to?",
  "model": "gpt-3.5-turbo-0125",
  "context_directory": "grid_applications/distribution_grid/grid_dss/src/systems/123Bus"
}'
data:

Final Summary:


data: Load S1A is connected to Bus 1.1.

Load S1A is connected to bus 1.1.

The provided context does not mention a load named S1A in the electrical distribution system data. Therefore, there is no information available regarding the bus to which Load
S1A is connected.

data: [DONE]

(venv) tinngo@Tins-MacBook-Pro sddec24-20 %
```

This screen shot shows how changing the model changes the output and makes it more accurate.

```
(venv) tinngo@Tins-MacBook-Pro sddec24-20 % curl -X POST http://127.0.0.1:5000/generate_text \
-H "Content-Type: application/json" \
-d '{
  "prompt": "How much active power is consumed by Load S1A?",
  "model": "gpt-3.5-turbo-0125",
  "context_directory": "grid_applications/distribution_grid/grid_dss/src/systems/123Bus"
}'
{
    "response": "The provided extracted information does not explicitly specify the active power consumed by Load S1A."
}%
(venv) tinngo@Tins-MacBook-Pro sddec24-20 % curl -X POST http://127.0.0.1:5000/generate_text \
-H "Content-Type: application/json" \
-d '{
  "prompt": "How much active power is consumed by Load S1A?",
  "model": "gpt-4-turbo-preview",
  "context_directory": "grid_applications/distribution_grid/grid_dss/src/systems/123Bus"
}'
{
    "response": "The active power consumed by Load S1A is 40.0 kW."
}%
(venv) tinngo@Tins-MacBook-Pro sddec24-20 %
```

3. Q: Is Load S1A a single phase or three phase load?

   A: Single phase

```
(venv) tinngo@tinngo sddec24-20 % curl -X POST http://127.0.0.1:5000/generate_text \
-H "Content-Type: application/json" \
-d '{
  "prompt": "Is Load S1A a single phase or three phase load?",
  "model": "gpt-3.5-turbo-0125",
  "context_directory": "grid_applications/distribution_grid/grid_dss/src/systems/123Bus"
}'
{
    "response": "The provided context does not contain specific information about whether Load S1A is a single-phase o
r three-phase load."
}%
(venv) tinngo@tinngo sddec24-20 % curl -X POST http://127.0.0.1:5000/generate_text \
-H "Content-Type: application/json" \
-d '{
  "prompt": "Is Load S1A a single phase or three phase load?",
  "model": "gpt-4-0125-preview",
  "context_directory": "grid_applications/distribution_grid/grid_dss/src/systems/123Bus"
}'
{
    "response": "Based on the provided extracted information, Load S1A is a single-phase load."
}%
```

My changes can be seen here in this Merge Request

https://git.ece.iastate.edu/sd/sddec24-20/-/merge_requests/13/diffs?view=parallel

Week 3:

While developing the ability to take in all the data given the DSS files to answer the operator's questions, I found that compiling all of the contexts from the DSS files was too much data for openAI. It had a few problems: It would not fit within the token context limit of the smaller models, it would simply be too much data to interpret, and a few more issues. So I developed a process where it would chunk up the data and summarize each file before trying to answer the question. This still had problems with openAI being unable to correctly find the data. My next solution would be to gather all of the files and their contexts, then ask openAI to select the files that it would need to answer the question, then selectively choose the files to answer the prompt. This is a process of multiple guided AI prompting and utilization to optimally find the answer given a question. This only works if we use a model that understands DSS data, filenames, etc. Here are screenshots of the outputs.

Here you can see the accuracy of loading all the DSS data as background information for openAI gpt-3.5-turbo-0125 to answer the question. You can see that it is not able to figure out the answer.

```
(venv) tinngo@tinngo sddec24-20 % curl -X POST http://127.0.0.1:5000/generate_text_openai_all_dss_files \
-H "Content-Type: application/json" \
-d '{
  "prompt": "Is Load S1A a single phase or three phase load?",
  "model": "gpt-3.5-turbo-0125",
  "context_directory": "grid_applications/distribution_grid/grid_dss/src/systems/123Bus"
}'
{
    "response": "The extracted information does not contain the specific details needed to determine if Load S1A is a single-p
hase or three-phase load."
}
```

In the selective algorithm, we can see improvements even though we are using the same GPT model. Here it is able to identify the answer given the prompt.

```
(venv) tinngo@tinngo sddec24-20 % curl -X POST http://127.0.0.1:5000/generate_text_openai_selective_dss_files \
-H "Content-Type: application/json" \
-d '{
  "prompt": "Is Load S1A a single phase or three phase load?",
  "model": "gpt-3.5-turbo-0125",
  "context_directory": "grid_applications/distribution_grid/grid_dss/src/systems/123Bus"
}'
{
    "response": "Based on the extracted information provided, Load S1a is a single-phase load. No further information is given
 to determine if it is a three-phase load."
}
```

This work has huge impact on our project because we will see marginal benefits from this. We are using less tokens which results in less time processing data, less money spent, and higher accuracy by providing a focused view for the LLM. I also write data to a bunch of files to show how they differ. One example is writing out the token usage. The selective data only uses 4634 tokens vs the unselective data which uses 37827 tokens. Here you can see the benefit. This is a 37827/4634 = 816% improvement. This will vary depending on the circumstances but in this case, we see huge benefits.

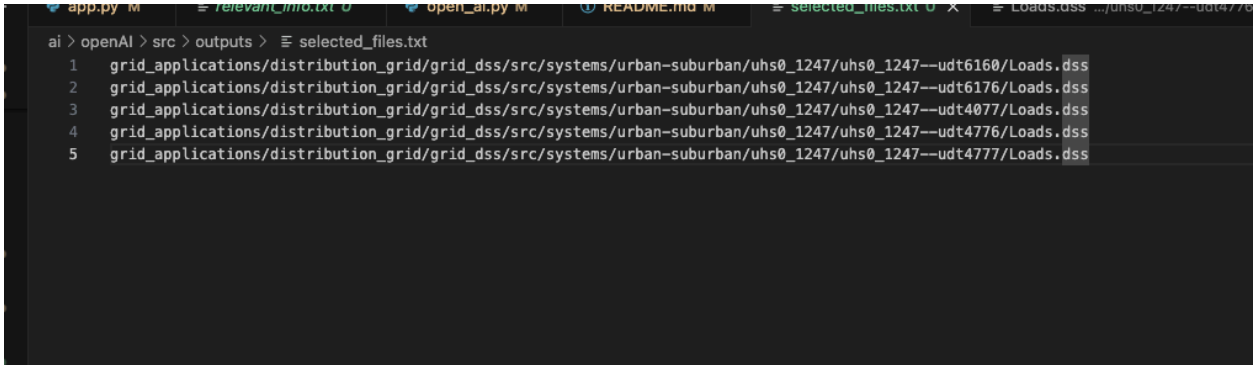
Week 4:

While working with Eddy on the front-end, we were setting up the endpoints and it was tough to do so because the backend to spin up the flask applications for openAI and huggingFace were not set up yet. I created Dockerfiles for both modules and connected it to the main system by adding it to docker-compose.yml. I had to restructure some files so that this would work. My work can be shown here. https://git.ece.iastate.edu/sd/sddec24-20/-/merge_requests/16/diffs

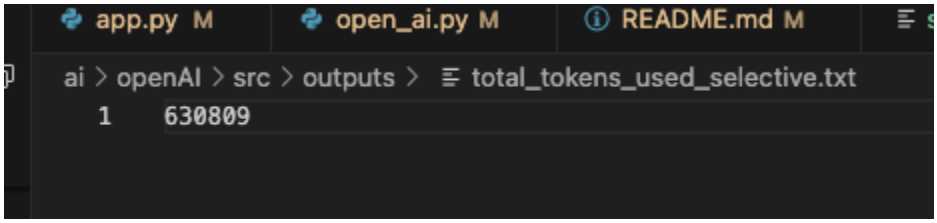
I also took Professor Gelli's feedback to make the system more resilient by handling filenames that may not match the conventional way of how dss files are structured. He said that I should look at a real dss model. I imported a real dss model from https://egriddata.org/group/national-renewable-energy-laboratory. I learned that it is a lot larger than the smaller subsets we were working with.

I then developed automation that looks for all dss files in all subdirectories and tries to identify which dss

file contains the contents to answer the prompt. What I found was to be less helpful than my previous implementation. The reason being is that there is too much data in a real dss model. There are multiple different dss files that have the same name like in the screenshot below, the system grabs 5/28 Loads.dss files to grab information from.



From the 5 it consumed an unhealthy amount of tokens:



It is not a good idea to look through subdirectories, instead, the front end should provide a specific directory.

There was an error on the front end side saying that the API was not allowed to be used so I worked with Eddy Andrade-Robles to enable my APIs by modifying the main.go file. I enabled three API endpoints. This was needed because main.go acts as a middleman between the front-end and the openAI flask app.

- Jackson Phillips:

  Week 1: This week I attempted to get a sample training job running on HPC. In order to do that I had to set up all the requirements for the huggingface libraries in a virtual environment on my VM and on HPC and read documentation on huggingface libraries and HPC. I have code for training but I am not entirely sure it functions as intended because I have run into errors every time I tried to run it. I wrote a batch script for slurm in order to run the job and the job ran but threw an error

saying transformers is not installed so I will need to figure out how to install all the libraries in a batch script. I also met with Tin and we talked about HPC and how slurm work and how to get the virtual environments set up. I am using gpt2 on huggingface solely for testing purposes because when I tried llama3 it said I don't have access to it. I am also using the IMDB dataset from huggingface because we do not have a prepared dataset as of now and this will still help me test my code and when we have our dataset I can swap it out.

```
#!/bin/bash
#SBATCH --job-name=jobtest1            # job name
#SBATCH --time=00:15:00                 # walltime limit (HH:MM:SS)
#SBATCH --nodes=1                       # number of nodes
#SBATCH --ntasks-per-node=36            # 36 processor core(s) per node
#SBATCH --mem=16G                       # maximum memory per node
#SBATCH --mail-user=jacksonp@iastate.edu  # email address
#SBATCH --mail-type=BEGIN
#SBATCH --mail-type=END
#SBATCH --mail-type=FAIL

# Load any necessary modules
module load python/3.8     # Adjust version as needed
# If you need specific libraries, load them here
# module load other_module   # Uncomment and adjust as necessary

# Set the working directory
cd /work/gelli/jacksonp

# Activate your virtual environment if needed
# source /path/to/your/venv/bin/activate  # Uncomment and adjust as necessary

# Run your Python script
python trainingtestcodejp1.py  # Replace with your actual Python script name
```

```python
# Use a pipeline as a high-level helper
from transformers import pipeline
from datasets import load_dataset
from transformers import AutoTokenizer, AutoModelForCausalLM
from trl import SFTTrainer
from transformers import TrainingArguments
import torch

# Ensure your model is on the CPU
device = torch.device("cpu")

# Load the tokenizer and model
tokenizer = AutoTokenizer.from_pretrained("openai-community/gpt2")
# Set the padding token to be the same as the end-of-sequence token
tokenizer.pad_token = tokenizer.eos_token

model = AutoModelForCausalLM.from_pretrained("openai-community/gpt2")
model.to(device)
# Load dataset
ds = load_dataset("stanfordnlp/imdb", split="train")

# Define training arguments
training_args = TrainingArguments(
    per_device_train_batch_size=2,
    gradient_accumulation_steps=4,
    warmup_steps=5,
    max_steps=60,
    learning_rate=2e-4,
    logging_steps=1,
    optim="adamw_8bit",
    weight_decay=0.01,
    lr_scheduler_type="linear",
    seed=3407,
    output_dir="outputs",
    save_steps=10,  # Save the model every 10 steps
    save_total_limit=2,  # Keep only the last 2 saved models
    logging_dir='logs',  # Directory for storing logs
)

```

```python
# Initialize the trainer
trainer = SFTTrainer(
    model=model,
    tokenizer=tokenizer,
    train_dataset=ds,
    dataset_text_field="text",
    dataset_num_proc=2,
    packing=False,  # Can make training 5x faster for short sequences.
    args=training_args,
)

# Start training
trainer.train()

# Save the trained model
trainer.save_model("outputs/final_model")

# Output training results
print("Training complete!")
print(f"Model saved to: outputs/final_model")
```

Week 2

```
Enter a prompt (or type 'exit' to quit): hello
The attention mask and the pad token id were not set. As a consequence, you may observe
 unexpected behavior. Please pass your input's `attention_mask` to obtain reliable resu
lts.
Setting `pad_token_id` to `eos_token_id`:50256 for open-end generation.
The attention mask is not set and cannot be inferred from input because pad token is sa
me as eos token. As a consequence, you may observe unexpected behavior. Please pass you
r input's `attention_mask` to obtain reliable results.

Generated Text: hello, I'm sorry to say that I'm not a fan of the movie. I'm not a fan
of the movie. I'm not a fan of the movie. I'm not a fan of the movie. I'm not a fan of
the movie. I'm not a fan of the movie. I'm not a fan of the movie. I'm not a fan of the
 movie. I'm not a fan of the movie. I'm not a fan of the movie. I'm
```

```
Enter a prompt (or type 'exit' to quit): star wars

Generated Text: star wars, and the war of words between two people
. The film is a good example that this movie can be made to work o
n any level.<br /><b />The story starts with an old man who has be
en living in his house for

Enter a prompt (or type 'exit' to quit): lord of the rings

Generated Text: lord of the rings, and a very good actor. He is al
so one to watch for his role in "The Last Man on Earth". The film
is well made with great acting by Michael Caine (who plays Mr Robo
t) as an old man

Enter a prompt (or type 'exit' to quit): tom cruise

Generated Text: tom cruise ship, the USS Enterprise-D. The crew of
 this vessel is a group that includes Captain Kirk (James T. Kirk)
, Commander Spock and Dr McCoy as well; they are all members from
the original Star Trek series which was released in

Enter a prompt (or type 'exit' to quit):
```

```
Enter a prompt (or type 'exit' to quit): iowa state

Generated Text: iowa state in southwestern India. The story of the Indian-Indian War is a fascinating historical film about four brave soldiers who were both killed and captured by a fanatic group on topo
graphical grounds; they must work together to save their own life after being defeated at gunpoint as part (or more)of an attempt launched against them during Operation Independence Day, August 1. Their h
eroic deeds are explained convincingly by this brilliant documentary director Jeevan Ramanamurthy with his masterful cinematography that conveys many shades: one day we will all feel it too when I see som
eone's bravery made famous through a cinema - if you watch 'Shoot Me In Your Nog' before me and remember how much joy my son took watching war movies back then or listen afterwards for any reason whatsoev
er...we'll soon come around to what happens next! A movie whose performance is only surpassed even by some Westerners such Asimath Sinha once he was shown off again because "I felt like living". It is sad
 enough, but still there has been so little recognition given our protagonists since 1947 from either side.<br /><br />While viewing a piece written here recently entitled ''Is My Dad Afraid Of Dying?'' I
 had always assumed him thinking he would never have died two years ago unless he just stopped talking now!!<span style="width:"100%" class="" color="#f4efe8"][/size]The ending should not be forgotten how
ever its almost laughable considering her death did nothing to make us smile.</spoiler>With this in mind check out the script which begins with Raju Tharoor saying that people do realise your parents grew
 up playing cricket firstly towards puberty whilst everyone else grows older until age 30 while getting into stardom due primarily mainly education followed along with training every year till old adult b
ecomes engaged. But why am i surprised?? Do these children get married early where later marriage makes kids start marrying young adults instead?"It seems quite surprising she decided to marry another man
....that sounds totally natural.....she knew exactly How wrong those women'd become."Wow..if only men could learn things quicker", though maybe husbands weren't going bald sooner rather than earlier!!! Or
 does tharpon ever actually die alone??? Why can anyone possibly take away anything? <b/c donut pics give rise mostly random killings.)Another interesting aspect regarding Rashmi Gandhi came forward today
 making no mention except very briefly describing herself--this time having seen films twice involving

Enter a prompt (or type 'exit' to quit): exit
```

This week I trained OpenAI's GPT2 model from hugging face using the IMDB dataset also from hugging face. I also wrote a python file to experiment with the trained model. I was getting repetitive responses at the beginning but I gave it a repetition penalty and it started giving responses that made more sense.

trainingtestcodejp1.py M | run_model.py U | jobtest1.sh M | requirements.txt

ai > huggingFace > src > run_model.py > ...

```python
from transformers import AutoModelForCausalLM, AutoTokenizer

# Load the fine-tuned model and tokenizer
model = AutoModelForCausalLM.from_pretrained("outputs/trained_model1")
tokenizer = AutoTokenizer.from_pretrained("outputs/trained_model1")

# Ensure the model is in evaluation mode
model.eval()

while True:
    prompt = input("Enter a prompt (or type 'exit' to quit): ")
    if prompt.lower() == 'exit':
        break

    # Encode input prompt
    inputs = tokenizer(prompt, return_tensors="pt", padding=True)

    # Generate text with adjusted parameters
    outputs = model.generate(
        inputs["input_ids"],
        attention_mask=inputs["attention_mask"],
        pad_token_id=tokenizer.eos_token_id,  # Ensure pad token is set
        min_length=2,
        max_length=200,           # Shorten the length to reduce repetition
        temperature=0.9,         # More randomness in output
        top_p=0.9,               # Nucleus sampling
        top_k=50,                # Top-k sampling
        repetition_penalty=1.5,    # Penalize repetition
        do_sample=True,
    )

    # Decode and print the output
    generated_text = tokenizer.decode(outputs[0], skip_special_tokens=True)
    print(f"\nGenerated Text: {generated_text}\n")
```

Week 3:

This week due to not having dataset access I have not been able to make much progress on training. I had some issues with my /home directory filling up because of python virtual environments and I emailed hpc and they sent me some resources to get it resolved but I had to spend a couple of hours reconfiguring my work on hpc so I can continue to use it correctly.

The team also met last week and I helped with some research on the script for generating dss files.

Week 4: This week I spent most of my time trying to get an untrained llama 3 model running on my

VM with the help of Tin's huggingface.py and the flask app. I also wrote a small script to allow me to prompt the model with user input without the flask app for my own testing.

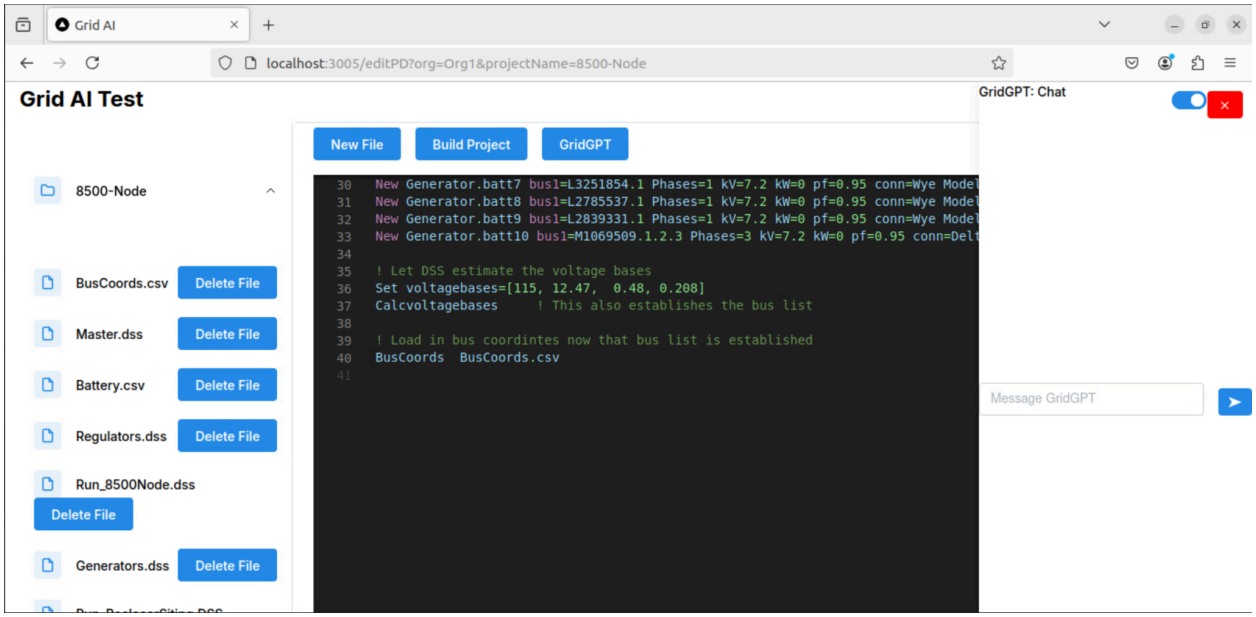
```
huggingFace > src > test_huggingFace.py > ...
from huggingFace import TextGenerator

if __name__ == "__main__":
    # Create an instance of TextGenerator
    text_generator = TextGenerator()

        # Take user input for the prompt
    prompt = input("Enter your prompt: ")

    max_length = int(input("Enter the maximum length of the generated text (defaul

    # Generate text
    generated_text = text_generator.generate_text(prompt, max_length=max_length)

    # Print the generated text
    print("\nGenerated Text:\n")
    print(generated_text)
```

I am also starting to work on file read/write but I am pretty sure that is not working yet because I have not had time to test it properly.

```
ai > huggingFace > src > file_test_huggingFace.py > ...
import os
import glob
from huggingFace import TextGenerator

# Load the TextGenerator model globally
text_generator = TextGenerator()

def edit_dss_files(directory, prompt):
    """Edit all .dss files in the specified directory based on the user's prompt."""
    files = glob.glob(os.path.join(directory, '*.dss'))
    for file_path in files:
        with open(file_path, 'r+') as file:
            content = file.read()
            # Here, you can modify the content based on the prompt
            # For example, let's assume we want to append generated text to each file
            generated_text = text_generator.generate_text(prompt)
            file.write(f"\n{generated_text}\n")  # Append generated text to the file

if __name__ == "__main__":
    input_directory = input("Enter the directory containing .dss files: ")

    while True:
        prompt = input("Enter your prompt (or type 'exit' to quit): ")
        if prompt.lower() == 'exit':
            break

        # Edit the .dss files based on the user's prompt
        edit_dss_files(input_directory, prompt)
        print("Files updated based on your prompt.")
```

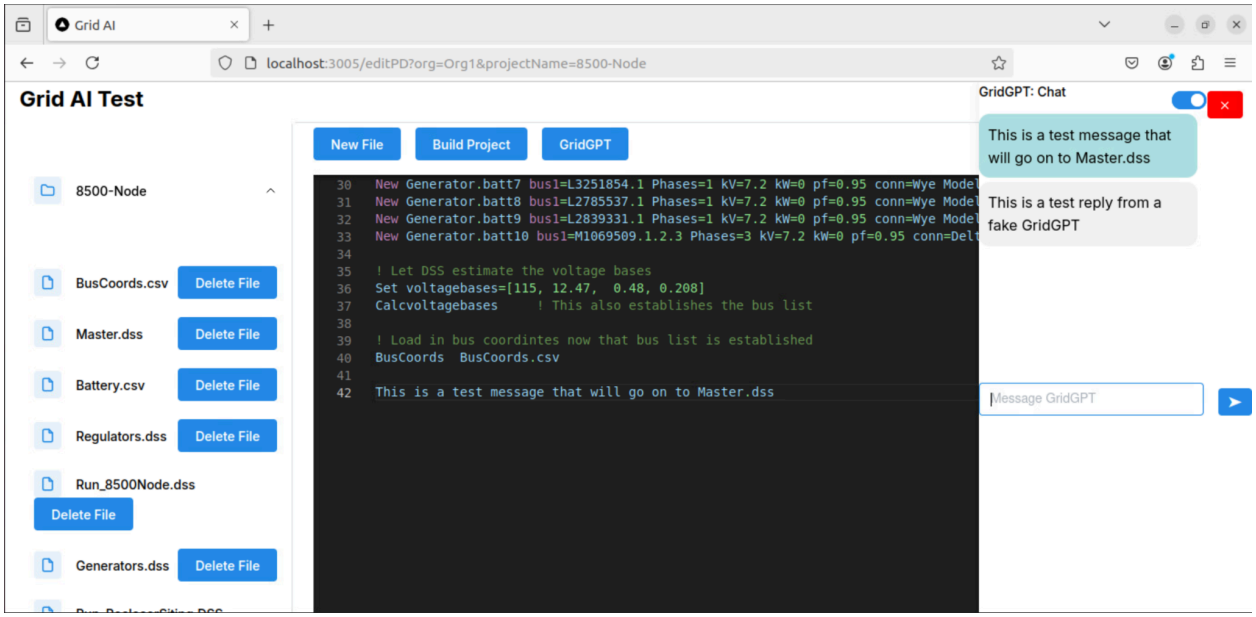
- Emma Heithoff: During week 1, Tin and I met to clarify the input output pairs the AltDSS-Python package ran simulations will produce numerical data for. The numerical data will be analyzed by the package in order to give examples for the input and output pair categories. In the following weeks, I attempted to determine whether a grid model, in a .dss file format, was a stable system by using AltDSS to give an error if the voltages at any buses were not within reasonable limits. I created a week 10 milestone to be finished with a dataset of input-output pairs. As I tried to fix the errors, I realized that I couldn't remember enough background on the AltDSS coding process I previously knew. I spoke with Professor Fila and Professor Gelli on concerns and the changes I needed to make for my weekly contribution to create a dataset. I spoke with Nick on his contribution to the dataset and how we will work together going forward. The milestones and their adjusted timelines for my situation will be sent to Professor Gelli soon. I will attach screenshots of my debugging work in the next weekly report when it is possible personally.

- Eddy Andrade: I created the chat window and followed the layout similarly to how it was portrayed in the Design Document. In the first screenshot, it shows what the layout of the project looks like.

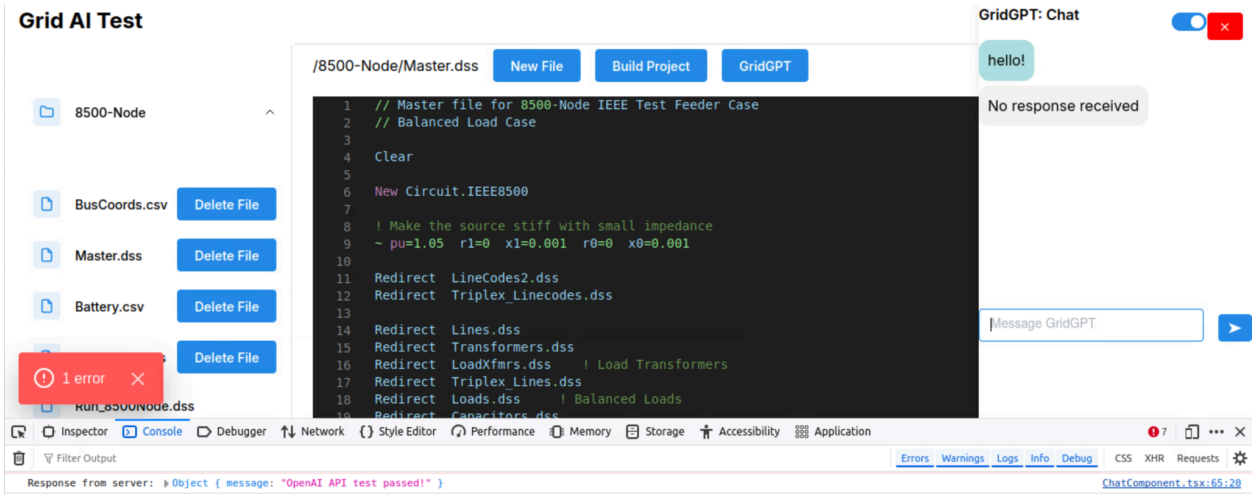


Alongside making the design, I began testing the chat feature. I created and modified code to allow the user to send a message between itself and the GridGPT chat bot. At the time of this screenshot, it only sent a hard-coded message as a response. I also began testing how messages

in the chat can be printed on the selected file in the code editor. This way, when GridGPT makes a suggestion on how to change the code on the file selected, it can make changes from the chat window into the file.



Finally, me and Tin began working on making an endpoint call to OpenAI's ChatGPT through an API call. We first began by using a test endpoint to make sure there is a connection, and the endpoint can receive the requests made. I changed how GridGPT would send a message, as it will now be able to send messages to the user instead of having a hard-coded message. At first, we had issues where the directory was posting an empty string, so I began troubleshooting and figuring out what the problem was. I then figured out what was causing the issue, and resolved it to now show the working directory even when a file is clicked. This way, we can use it in the request body when making that API call to OpenAI.



In this screenshot, you can see how the hello message was received by OpenAI's server. The GridGPT response says "No response received", but that's because the success message was sent only through the console and not back to the user as an actual response. More on the popup error in the "Pending Issues" category.

- Nick Doty: I have done research on how to generate question and answer pairs with altdss and have experimented with code that can be utilized to run a powerflow simulation and then generate question/ answer pairs from a given dss file. In particular, I have been working on one that can generate question and answer pairs about the stability of transformers.

  Shown below was my initial interpretation of the datasets in JSON format.

```
data:
  0:
    timestamp:      "2024-10-01T00:00:00Z"
    voltage:        230
    current:        100
    power_output:   23000
    status:         "stable"
  1:
    timestamp:      "2024-10-01T01:00:00Z"
    voltage:        228
    current:        102
    power_output:   23256
    status:         "stable"
  2:
    timestamp:      "2024-10-01T02:00:00Z"
    voltage:        220
    current:        110
    power_output:   24200
    status:         "stable"
  3:
    timestamp:      "2024-10-01T03:00:00Z"
    voltage:        215
    current:        115
    power_output:   24725
    status:         "unstable"
  4:
    timestamp:      "2024-10-01T04:00:00Z"
    voltage:        210
    current:        120
    power_output:   25200
    status:         "unstable"
  5:
    timestamp:      "2024-10-01T05:00:00Z"
    voltage:        205
    current:        125
    power_output:   25625
    status:         "unstable"
```

```
  6:
    timestamp:      "2024-10-01T06:00:00Z"
    voltage:        200
    current:        130
    power_output:   26000
    status:         "critical"
  7:
    timestamp:      "2024-10-01T07:00:00Z"
    voltage:        195
    current:        135
    power_output:   26325
    status:         "critical"
  8:
    timestamp:      "2024-10-01T08:00:00Z"
    voltage:        190
    current:        140
    power_output:   26600
    status:         "failure"
```


Generator instability:

```
▼ power_line:
    id:                "PL-001"
    location:          "Substation A"
  ▼ data:
    ▼ 0:
        timestamp:     "2024-10-01T00:00:00Z"
        voltage:       230
        current:       50
        power:         11500
        status:        "normal"
    ▼ 1:
        timestamp:     "2024-10-01T01:00:00Z"
        voltage:       230
        current:       55
        power:         12650
        status:        "normal"
    ▼ 2:
        timestamp:     "2024-10-01T02:00:00Z"
        voltage:       230
        current:       60
        power:         13800
        status:        "normal"
    ▼ 3:
        timestamp:     "2024-10-01T03:00:00Z"
        voltage:       230
        current:       70
        power:         16100
        status:        "warning"
    ▼ 4:
        timestamp:     "2024-10-01T04:00:00Z"
        voltage:       230
        current:       80
        power:         18400
        status:        "warning"
    ▼ 5:
        timestamp:     "2024-10-01T05:00:00Z"
        voltage:       230
        current:       90
        power:         20700
        status:        "critical"
```

```
    ▼ 6:
        timestamp:     "2024-10-01T06:00:00Z"
        voltage:       230
        current:       100
        power:         23000
        status:        "critical"
    ▼ 7:
        timestamp:     "2024-10-01T07:00:00Z"
        voltage:       230
        current:       110
        power:         25300
        status:        "overloaded"
    ▼ 8:
        timestamp:     "2024-10-01T08:00:00Z"
        voltage:       230
        current:       120
        power:         27600
        status:        "overloaded"
    ▼ 9:
        timestamp:     "2024-10-01T09:00:00Z"
        voltage:       230
        current:       130
        power:         29900
        status:        "overloaded"
```

Powerline overload:


```
power_grid:
  location:          "Grid_A"
  data:
    0:
      timestamp:     "2024-10-01T00:00:00Z"
      voltage:       230
      current:       150
      temperature:   75
      thermal_limit: 200
      status:        "stable"
    1:
      timestamp:     "2024-10-01T01:00:00Z"
      voltage:       229.5
      current:       160
      temperature:   77
      thermal_limit: 200
      status:        "stable"
    2:
      timestamp:     "2024-10-01T02:00:00Z"
      voltage:       229
      current:       170
      temperature:   80
      thermal_limit: 200
      status:        "stable"
    3:
      timestamp:     "2024-10-01T03:00:00Z"
      voltage:       228.5
      current:       180
      temperature:   82
      thermal_limit: 200
      status:        "approaching_limit"
    4:
      timestamp:     "2024-10-01T04:00:00Z"
      voltage:       228
      current:       190
      temperature:   85
      thermal_limit: 200
      status:        "at_limit"
```

```
    5:
      timestamp:     "2024-10-01T05:00:00Z"
      voltage:       227.5
      current:       200
      temperature:   90
      thermal_limit: 200
      status:        "exceeded_limit"
    6:
      timestamp:     "2024-10-01T06:00:00Z"
      voltage:       227
      current:       210
      temperature:   92
      thermal_limit: 200
      status:        "critical"
    7:
      timestamp:     "2024-10-01T07:00:00Z"
      voltage:       226.5
      current:       220
      temperature:   95
      thermal_limit: 200
      status:        "restored"
    8:
      timestamp:     "2024-10-01T08:00:00Z"
      voltage:       230
      current:       140
      temperature:   70
      thermal_limit: 200
      status:        "stable"
```

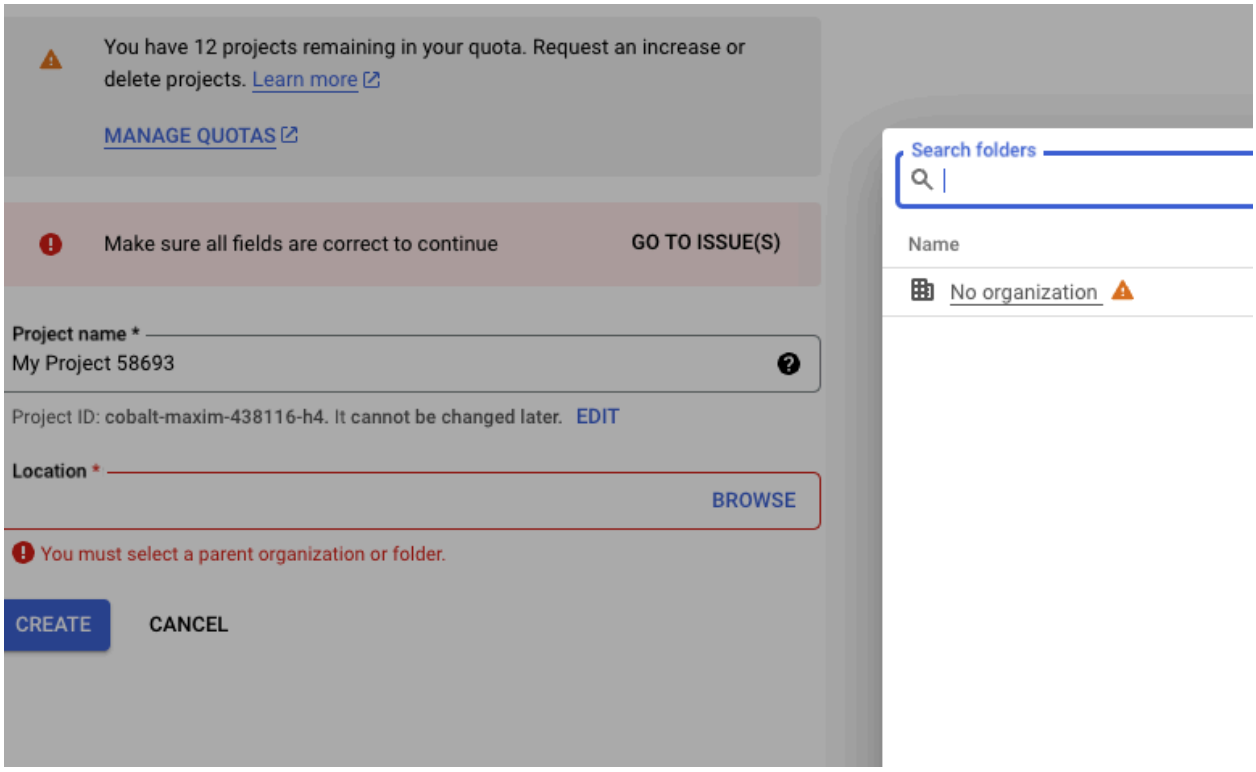
Below is the code I am attempting to work with based on the stability of transformers to generate question and answer pairs:

```python
import os
import json
import subprocess
from altdss import DSS

def run_simulation(dss_file):
    """Run the power flow simulation on a given DSS file."""
    try:
        dss = DSS()
        dss.load(dss_file)
        results = dss.run_power_flow()
        return results
    except Exception as e:
        print(f"Error running simulation on {dss_file}: {e}")
        return None

def analyze_results(results):
    """Analyze the simulation results to determine stability."""
    stable = True  # Default assumption
    transformer_power = {}

    for transformer in results['transformers']:
        if transformer['power'] > transformer['capacity']:
            stable = False
        transformer_power[transformer['name']] = transformer['power']

    return stable, transformer_power

def generate_qa_pairs(dss_file, stable, transformer_power):
    """Generate question and answer pairs based on the simulation results."""
    qa_pairs = []
```

```python
    for transformer, power in transformer_power.items():
        question = f"What is the power output of transformer '{transformer}' in the simulation with {dss_file}?"
        answer = "Stable" if stable else "Unstable"
        qa_pairs.append({"question": question, "answer": answer})

    return qa_pairs

def main(dss_directory):
    """Main function to process DSS files and generate QA pairs."""
    all_qa_pairs = []

    for filename in os.listdir(dss_directory):
        if filename.endswith('.dss'):
            dss_file_path = os.path.join(dss_directory, filename)
            results = run_simulation(dss_file_path)

            if results:
                stable, transformer_power = analyze_results(results)
                qa_pairs = generate_qa_pairs(dss_file_path, stable, transformer_power)
                all_qa_pairs.extend(qa_pairs)

    # Save the generated QA pairs to a JSON file
    with open('qa_pairs.json', 'w') as f:
        json.dump(all_qa_pairs, f, indent=4)

if __name__ == "__main__":
    dss_directory = 'path_to_your_dss_files'  # Update this path
    main(dss_directory)
```

- **Pending issues**

- Tin Ngo: Currently don't have access to GCP which is the cloud platform we plan on hosting our LLM. I tried creating a project in by myself to get started with testing but I could not



- Jackson Phillips: I am not able to copy files into the /work/gelli/sddec24-20 folder so I am using /work/gelli/jacksonp for the time being.

  I have some errors from my attempts to run the training test.

  This was the output file of the slurm job and I will look into how to install libraries and set up the virtual environment through the batch script.

```
Traceback (most recent call last):
  File "trainingtestcodejp1.py", line 49, in <module>
    from transformers import pipeline
ModuleNotFoundError: No module named 'transformers'
```

  I installed bitsandbytes because I ran the code and it said it posted an error saying it was needed but I am not sure how to get it compiles with GPU support on HPC.

```
The installed version of bitsandbytes was compiled without GPU support. 8-bit op
timizers, 8-bit multiplication, and GPU quantization are unavailable.
```

I tried running the code without a slurm batch file and it reported that there are no NVIDIA drivers. I assume this is because I am not on the correct node.

```
RuntimeError: Found no NVIDIA driver on your system. Please check that you have
an NVIDIA GPU and installed a driver from http://www.nvidia.com/Download/index.a
spx
```

- Emma Heithoff: I am having issues creating the dataset because I need to focus on AltDSS documentation again. I did need to focus on other aspects of the dataset, such as overall purpose, and will get help from the rest of the team on the simulation code I am struggling to run. I will be reaching out for more dataset assistance until the end of the semester to avoid more roadblocks.
- Eddy Andrade: As seen in my screenshot, I am having some errors during runtime (TypeError), but I am in the process of resolving those issues. Previously, I had a significant amount of errors, but resolved most of them.
- Nick Doty: Developing proper code and finding the proper dataset can be quite tedious, often on the surface seeming like no results are being produced.

- **Individual contributions**

NAME	**Individual Contributions** *(Quick list of contributions. This should be short.)*	**Hours this week**	**HOURS cumulative**
Tin Ngo	GCP and CI/CD, Generating responses given prompt. Answers are working. Modify main.go file to enable API	42	157
Jackson Phillips	Training code, HPC, slurm	35	115
Emma Heithoff	Voltage stability simulation attempts, clarifying gaps in AltDSS understanding / altered dataset timeline	33	114
Eddy Andrade	Frontend development, troubleshooting, modifying existing code	31	111
Nick Doty	Dataset development	31	111

- **Plans for the upcoming week**
  - Tin Ngo: Work more with Emma to handle outputs of altDSS simulations.
  - Jackson Phillips: work on functionality of llama 3
  - Emma Heithoff: I will email Professor Gelli a revised timeline, work in person with Nick on running simulations for a dataset, and ask the rest of the team for assistance in the simulation code after Nick and I work on it.
  - Eddy Andrade: I will continue to work on the endpoints (more specifically, the JSON requests) in order to get a response with OpenAI's API call.
  - Nick Doty: Continue talking with Jackson to see what other datasets would be needed and how to tweak them to suit our needs.

- **Summary of weekly advisor meeting**